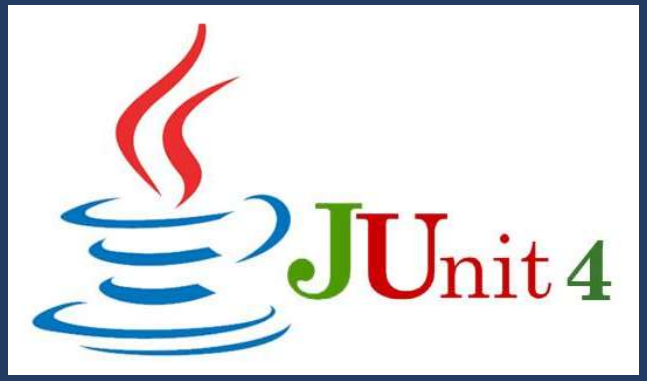
JUNT && ASSERTIONS



JUNIT

- JUnit est un framework open source pour le développement et l'exécution de tests unitaires automatisables. Le principal intérêt est de s'assurer que le code répond toujours aux besoins même après d'éventuelles modifications. Plus généralement, ce type de tests est appelé tests unitaires de non-régression.
- Le but est d'automatiser les tests. Ceux-ci sont exprimés dans des classes sous la forme de cas de tests avec leurs résultats attendus. JUnit exécute ces tests et les comparent avec ces résultats.
- JUnit permet de lancer facilement toutes les methodes marquees @Test
  - Maven les lance ensuite dans sa phase «test»
  - On utilise des assertions pour valider
- Les assertions servent a valider une instance d'objet ou une valeur

JUNIT : CONFIGURATION MAVEN

Ajouter la dependence:

```
<!-- https://mvnrepository.com/artifact/junit/junit -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.13.2</version>
    <scope>test</scope>
</dependency>
```

JUNIT : TEST FIXTURE

JUnit fournit des des annotations pour une création de test fixture:

Annotations	La description
@Test	Cette annotation remplace org.junit.TestCase qui indique que la méthode void publique à laquelle elle est attachée peut être exécutée en tant que scénario de test.
@Before	Cette annotation est utilisée si vous souhaitez exécuter une instruction telle que les conditions préalables avant chaque scénario de test.
@BeforeClass	Cette annotation est utilisée si vous souhaitez exécuter certaines instructions avant que tous les cas de test, par exemple la connexion de test, doivent être exécutés avant tous les cas de test.
@After	Cette annotation peut être utilisée si vous souhaitez exécuter certaines instructions après chaque scénario de test, par exemple pour réinitialiser des variables, supprimer des fichiers temporaires, des variables, etc.
@AfterClass	Cette annotation peut être utilisée si vous souhaitez exécuter certaines instructions après tous les cas de test, par exemple pour libérer des ressources après avoir exécuté tous les cas de test.

JUNIT : ASSERTIONS

JUnit fournit des assertions de base (egalite, erreur, etc...)

Méthode	Rôle
assertEquals()	Vérifier l'égalité de deux valeurs de type primitif ou objet (en utilisant la méthode equals()). Il existe de nombreuses surcharges de cette méthode pour chaque type primitif, pour un objet de type Object et pour un objet de type String
assertTrue()	Vérifier que la valeur fournie en paramètre est vraie
assertFalse()	Vérifier que la valeur fournie en paramètre est fausse
assertNull()	Vérifier que l'objet fourni en paramètre soit null
assertNotNull()	Vérifier que l'objet fourni en paramètre ne soit pas null
assertSame()	Vérifier que les deux objets fournis en paramètre font référence à la même entité Exemples identiques : assertSame("Les deux objets sont identiques", obj1, obj2); assertTrue("Les deux objets sont identiques ", obj1 == obj2);
assertNotSame()	Vérifier que les deux objets fournis en paramètre ne font pas référence à la même entité

JUNIT VS TESTNG

- TestNG est inspiré de JUNIT qui utilise les annotations (@).
- TestNG et JUnit4 se ressemblent, à l'exception d'une ou deux fonctionnalités.
- TestNG est conçu pour faciliter les tests de bout en bout.
- Il permet des tests en parallèle.

	TestNG	JUNIT
Annotations	✓	✓
Suite de test	✓	✓
Ignorer le test	✓	✓
Test d'exception	✓	✓
Temps libre (Time out)	✓	✓
Test paramétré	✓	✓
Test de dépendance	✓	X